Secure HTTP Update

IETF WTS WG (18-Jul-95)

Presented By:

Allan M. Schiffman
Chief Technical Officer
Enterprise Integration Technologies Corp.
800 El Camino Real
Menlo Park, CA 94025
(415) 617-8005
ams@eit.com

Changes for July I-D

- More Motivating Material
- New Examples
- Clarifications
- Depreciated Facilities
    - PGP encapsulation format
    - PKCS-6 Certificates

S-HTTP: Design Goals & Features

- Enable Spontaneous Commercial Transactions
- Negotiation of Algorithms, Modes & Parameters
- Layer separation (Don't "Fix" HTTP)
- Mechanism, not Policy
- Interoperability
- Fully symmetric (almost) for client and server
- Orthogonal mechanisms: authentication, key exchange, encryption
    - Authentication: PKC (RSA, DSS), Shared-Secret, Kerberos
    - Key Exchange: PKC (RSA), Shared-Secret, Kerberos
- Impacts
    - New HTTP method ("Secure") -- affects proxies
    - New (S-)HTTP headers
    - Additional HTML facilities

S-HTTP Security Encapsulation

- Uses Existing (Security) Encapsulation Formats
  - Uses Most PKCS-7 types:
    - Data, SignedData, EnvelopedData, SignedAndEnvelopedData, EncryptedData
    - Doesn’t use DigestedData
  - Adds new facilities for keyed-digest MACs and symmetric key-management

S-HTTP Focus: Negotiation

- Permits Parties to Express Requirements and Preferences.
- Used in Message Headers and Embedded in Documents
- Choice May Depend on:
    - Capability of Implementation
    - Application Requirements
- Schema
    - Property, such as “Bulk Encryption Algorithm”
    - Value, such as “DES-CBC”
    - Direction: Receive or Originate
    - Strength: Required, Optional, Refused
    - “*You may use DES-CBC when encrypting.*”

S-HTTP Focus: Negotiable Items

- Encapsulation Format: PKCS-7, PEM
- Signature Algorithm: RSA or DSA
- Key Exchange Algorithm:
  RSA, In-band, “Outband”, Kerberos
- Message Digest Algorithm: MD2, MD5 or SHA
- Encryption Algorithms:
  DES, DES-EDE2/3, DESX, CDMF, IDEA, RC2, RC4
- Protection Mode: Signature, Encryption, Keyed MAC
- Certificate Pattern (e.g., “Mastercard or Visa”)

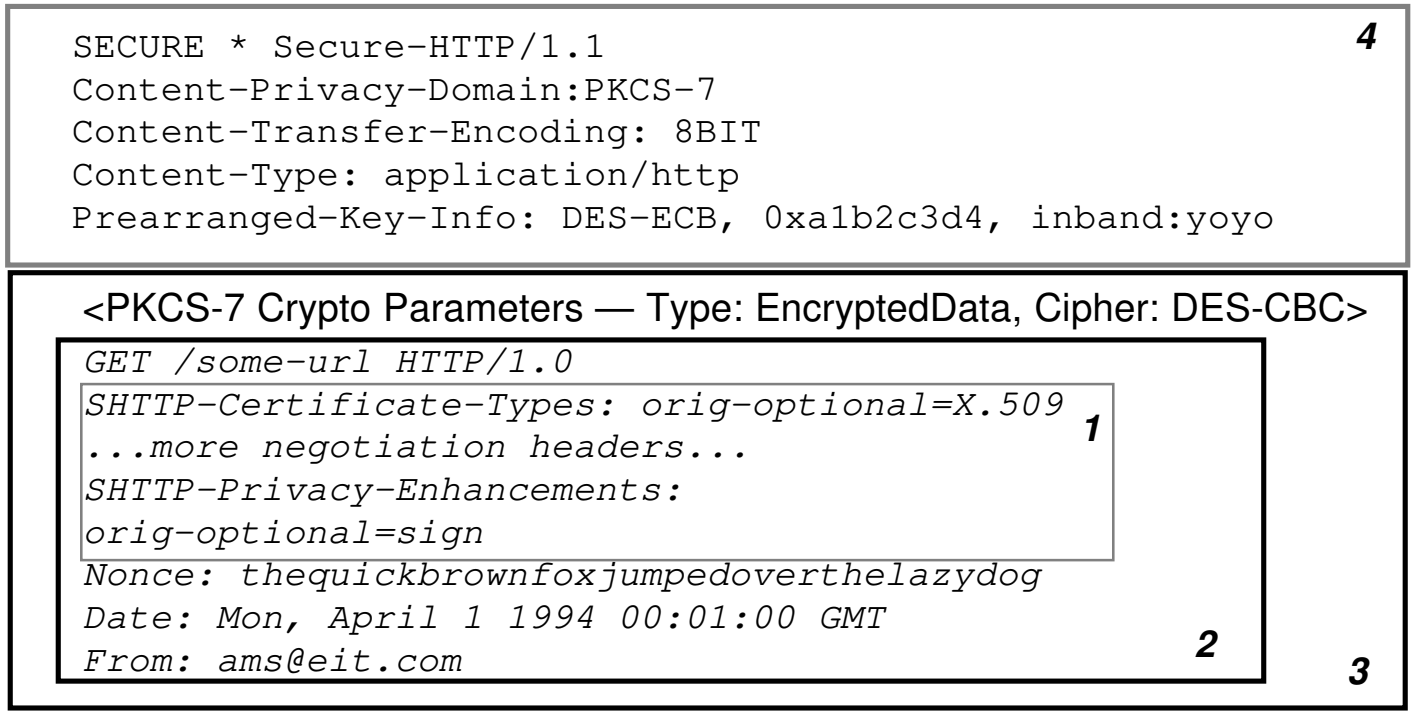
S-HTTP Negotiation Example

A Common Case

- Server Specifies:
  - Client Must Encrypt
  - Server Willing to Sign and/or Encrypt
  - Transaction Key Exchanged Under...
    - Server's Public Key when Client Encrypts
    - Server or Client's PK when Server Encrypts

```
SHTTP-Key-Exchange-Algorithms:      recv-required=RSA;
                                    orig-optional=Inband, RSA
SHTTP-Privacy-Enhancements:         orig-optional=sign, encrypt;
                                    recv-required=encrypt
```

One S-HTTP Message, Illuminated



1. SHTTP Negotiation Headers
2. HTTP Message (Encrypted with DES-CBC)
3. PKCS-7 Encapsulated Message (ASN.1 Encoded)
4. Outer S-HTTP Headers

S-HTTP (Unencapsulated) headers

- Content-Privacy-Domain: {PKCS-7, PEM}
- Content-Transfer-Encoding: {8BIT, 7BIT, BASE64, BINARY}
- Content-Type: application/http
-
- Prearranged-Key-Info: <Hdr-Cipher>, <Key>, <Key-ID>
- MAC-Info: [hex(TOD),] <hash-alg>, hex(hash-data), <key-id>

HTTP (Encapsulated) Non-negotiation Headers

- Security-Scheme: S-HTTP/1.1
- Key-Assign: <method>,<key-name>,<lifetime>,<ciphers>;<method-args>
- Encryption-Identity: <name-class>, <key-sel>, <name-arg>
- Certificate-Info: <Cert-Fmt>’,’<Cert-Group>
- Nonce: <string>
- Nonce-Echo: <string>

Encapsulated Negotiation Headers

- SHTTP-Privacy-Domains: {PKCS-7, PEM}
- SHTTP-Privacy-Enhancements:{sign, encrypt, auth}
- SHTTP-Certificate-Types: {X.509}
- SHTTP-Key-Exchange-Algorithms: {RSA, Outband, Inband, Krb-{4,5}}
- SHTTP-Signature-Algorithms: {RSA, NIST-DSS}
- SHTTP-Authentication-Algorithms: {MAC, Kerb-{4,5}}
- SHTTP-Message-Digest-Algorithms: {RSA-MD2, RSA-MD5, NIST-SHS}

Encapsulated Negotiation Headers (continued)

- SHTTP-Symmetric-Header-Algorithms:
  {DES-ECB, DES-EDE-ECB, DES-EDE3-EBC,
  DESX-ECB,CDMF-ECB, IDEA-ECB, RC2-ECB}
- SHTTP-Symmetric-Content-Algorithms:
  {DES-CBC, DES-EDE-CBC, DES-EDE3-CBC,
  DESX-CBC,CDMF-CBC, IDEA-CFB, RC2-CBC, RC4}
- DN-Pattern: <cchain-regexp>
- Your-Key-Pattern: {MAC, Kerberos, Outband}, <regexp>

HTML Extensions

- Anchor Attributes
  - DN
  - NONCE
  - CRYPTOPTS
- New HTML Elements
  - CERTS
  - CRYPTOPTS

HTTP Statelessness vs. Session Context

- HTTP “Statelessness” Prejudice Misguided
  - Connection Retention Needed for Performance
  - Many WWW Applications Need Session Threading
    - Defining Session Boundaries in HTTP is Hard
  - Forming Security Associations is Expensive
    - Especially When Using PKC
- S-HTTP Addresses the Latter
  - Inband Keys
  - Nonces
  - Implementation can Avoid Explicit State

Channel vs. Message Security

Or, Security in the Application Layer vs. Below

- Long History of Debate
    - Fought at every layer of the OSI model!
    - Active traditions, active future
        - Message/Object: PEM to MOSS
            - Pro: Most Meaningful & Flexible
            - Con: Must repeatedly reinvent
        - Network/Transport: SWiPE to IPSP
            - Pro: Simple & General, Fewer Errors
            - Con: Security Weaknesses, Users uninformed
- Postal Service metaphor: pouches in locked trucks?